An Introduction to Bootstrap Methods using *Arc*

Iain Pardoe
Department of Applied Statistics
School of Statistics
University of Minnesota, St. Paul, MN 55108

Technical Report Number 631
Work Supported by the National Science Foundation, Grant DUE 96-52887

February 24, 2000

Abstract

This report presents (1) the basic ideas of bootstrapping when applied to multiple linear regression, as described in [2, 3], and (2) how to implement these ideas using *Arc*, the computer package that accompanies [1].

1 Introduction

This report provides (1) an introduction to bootstrap methods in linear regression analyses as discussed in [2, 3], and (2) computer code for use with the program *Arc*, described in [1], that implements these analyses. The remainder of this section outlines the general ideas behind linear regression. In Section 2, I summarize the basic bootstrap approach to statistical inference, and present two ways of applying it to linear regression. Next, I describe some of the issues involved with estimating and testing mean function coefficients in Sections 3 and 4 respectively. Finally, I present some examples in Section 5, and outline some of the other areas in regression where bootstrap methods could be used to good effect in Section 6.

The most recent version of *Arc* can be obtained on the Internet from the link http://www.stat.umn.edu/arc. The *Xlisp-Stat* code to perform all the analyses in this report are given in the file http://www.stat.umn.edu/arc/boot.lsp. The commands needed to run this code are scattered throughout this report itself.

To use `boot.lsp`, download it from the Internet, and put it in the folder `Extras` in your `Arc` directory.

1.1 Who should read this report

In this report I describe additions to *Arc* that can be used to perform bootstrap sampling for the specific problems discussed. As these may have interest in various situations, many readers will find the additions useful. The additions can also provide a starting point for implementation of the bootstrap in other situations, but for this the reader must be able to read and write computer programs in the language *lisp*. There are several ways of getting started. Tierney in [4] provides a very readable introduction. Several on-line references can be obtained from http://www.visualstats.org/. The book *Common Lisp* by Guy L. Steele is a comprehensive source for the *lisp* language, and is available on-line at

http://www-cgi.cs.cmu.edu/afs/cs.cmu.edu/project/ai-repository/ai/html/cltl/clm/clm.html

A companion report on "Simulations using *Arc*" may be helpful, and is available from

http://www.stat.umn.edu/arc/simulate.pdf.

1.2 Linear regression

Following the notation and development in [1], regression concerns a *response* y and p *predictors*, $\mathbf{x} = (x_1, \ldots, x_p)^T$. The general goal in regression is to study how the conditional distribution of $y|\mathbf{x}$ changes as the value of $\mathbf{x}$ changes, often concentrating on the mean function, $\mathrm{E}(y|\mathbf{x})$. In many regression problems, the response $y|\mathbf{x}$ is written

$$y|\mathbf{x} = \mathrm{E}(y|\mathbf{x}) + e/\sqrt{w}$$

where e is called the statistical *error* and the weights $w > 0$ are *known, positive* numbers. Another feature of the conditional distribution of $y|\mathbf{x}$ that is often studied in regression is the variance function $\mathrm{Var}(y|\mathbf{x}) = \mathrm{Var}(e)/w$.

Let $\mathbf{u}$ be a $k \times 1$ vector of *terms* derived from $\mathbf{x}$. Typically, $\mathbf{u}$ will consist of a constant 1 for an intercept, and $(k-1)$ additional functions of $\mathbf{x}$, like polynomials or other transformations. The *linear regression model* has mean function

$$\mathrm{E}(y|\mathbf{x}) = \mathrm{E}(y|\mathbf{u}) = \eta_0 + \eta_1 u_1 + \cdots \eta_{k-1} u_{k-1} = \boldsymbol{\eta}^T\mathbf{u} \tag{1}$$

where $\boldsymbol{\eta}^T = (\eta_0, \ldots, \eta_{k-1})$ is a $k \times 1$ vector of mean function coefficients, and variance function

$$\mathrm{Var}(y|\mathbf{x}) = \sigma^2/w \tag{2}$$

These assumed forms of the mean and variance functions imply that $\mathrm{E}(e) = 0$ and $\mathrm{Var}(e) = \sigma^2$. This reflects an alternative way of specifying the general form of the linear regression model—the linear mean function (1) together with the assumption that the distribution of the errors is independent of $\mathbf{x}$.

For a full parametric analysis, the distribution of $y|\mathbf{x}$, or alternatively of e, must be specified. For normally distributed errors, the least squares theory of regression estimation and inference provides straightforward, exact methods for analysis. But for non-normal errors, these methods have the potential to be inaccurate or misleading. Resampling methods such as the bootstrap provide an alternative methodology, with the potential to both

- reinforce conclusions arrived at using normal theory, and

- to provide estimation and inference techniques in situations where normal theory does not seem to be justified.

The examples in this report focus mainly on the first of these goals, although Section 6 mentions some areas that could involve more in the way of the second of these goals.

2 Two alternative paradigms for using bootstrap methods in linear regression

The bootstrap is a data-based simulation method for statistical inference. The basic idea is as follows. I wish to make an inference about a (population) quantity, say θ, for which I have a data-based estimate, $\hat{\theta}$. I then want to get some idea of the distribution of my estimate, without having to make assumptions about my data (for example, that it comes from a multivariate normal distribution). One way to do this is to resample with replacement from my data to get a bootstrap sample (of the same size as my original sample, and made up of cases from my original sample, some appearing once, some twice, and so on, and some not appearing at all). I then create a large number, B, of such bootstrap samples, and calculate $\hat{\theta}$ for each sample. (For notation, I denote bootstrap estimates with a star, and hence $\hat{\theta}$ for a bootstrap sample is denoted $\hat{\theta}^*$.) These B $\hat{\theta}^*$'s contain information that can be used to make inferences from the data; essentially, $\hat{\theta}^*$ is to $\hat{\theta}$ as $\hat{\theta}$ is to θ. Some of the types of inference possible in the linear regression context will be explored in more detail in Sections 3 and 4.

For any particular application of the bootstrap, I need to decide on an appropriate way to resample with replacement from my data to get a bootstrap sample. For regression, I present two such ways, "resampling residuals" and "resampling cases."

2.1 Resampling residuals

The "resampling residuals" paradigm takes the point of view that responses, $y|\mathbf{x}$, are sampled from a univariate distribution $F(y|\mathbf{x})$ whose mean and variance are given by the mean function (1) and variance function (2). Trying to estimate $F(y|\mathbf{x})$ using bootstrap methods doesn't work directly, because of the conditioning on $\mathbf{x}$. But, since $\mathbf{x}$ is assumed to be fixed under this paradigm, I can instead try to estimate the distribution of e by resampling residuals, defined to be the weighted differences between the observed values of the response and the fitted values under the linear regression model

$$\hat{e} = \sqrt{w}(y - \hat{\mathrm{E}}(y|\mathbf{x})) = \sqrt{w}(y - \hat{y}) \tag{3}$$

The linear regression model is implicit under this paradigm—using bootstrap methods in regression by resampling residuals essentially assumes that the linear regression model holds.

2.2 Resampling cases

The "resampling cases" paradigm takes the point of view that cases, $(y, \mathbf{x})$, are sampled from a multivariate distribution $F(y, \mathbf{x})$. Application of the bootstrap is straightforward, and I can try to estimate the distribution of $(y, \mathbf{x})$ by resampling cases, defined to be the multivariate vectors $(y, \mathbf{x})$. The linear regression model lies outside this paradigm—using bootstrap methods in regression by resampling cases essentially makes no assumption about whether or not the linear regression model holds. The linear regression model only arises in this setting by going on to specify the conditional mean and variance of $y|\mathbf{x}$ as the mean function (1) and variance function (2). Then, it is also necessary to include each case's weight when resampling, and the cases to be resampled can now be written $(y, \mathbf{x}, w)$.

3 Estimating regression coefficients

3.1 Resampling residuals

As described above, I want to try to estimate the distribution of e by resampling residuals. However, since the distribution of e is assumed to be independent of $\mathbf{x}$, I really want to be resampling something whose distribution does not depend on $\mathbf{x}$. Assuming I have n cases in my data set, the i-th residual as defined in (3) has variance $\mathrm{Var}(\hat{e}_i|\mathbf{x}_i) = \sigma^2(1 - h_i)$, where h_i is the i-th *leverage*, $i = 1, \ldots, n$. The leverage depends on $\mathbf{x}_i$, and so if any leverages are particularly high, resampling the residuals will not do a good job of estimating the distribution of e. So, since the leverages are known, I consider instead the *modified residuals*

$$\hat{r}_i = \frac{\sqrt{w_i}(y_i - \hat{y}_i)}{\sqrt{(1 - h_i)}} \tag{4}$$

which have constant variance. The modified residuals differ from the *Studentized residuals* described in Section 15.3.1 of [1] by division by an estimate of σ^2, and thus the two are equivalent.

One final adjustment is needed since e is assumed to have mean 0. Thus, I actually sample from the mean-corrected modified residuals to get bootstrap errors

$$e_i^* \text{ randomly sampled with replacement from } \hat{r}_i - \bar{r}$$

where $\bar{r}$ is the sample average of the modified residuals (4).

I am interested in estimating regression coefficients, so I now need to derive bootstrap responses and predictor values from my bootstrap errors. Since, I am assuming that the linear regression model holds, I fix my bootstrap predictor values at the original sample $\mathbf{x}_i$ values (and I also fix my bootstrap weights at the original sample w_i values). I then use the usual weighted least squares estimate of $\boldsymbol{\eta}$ from the original sample, say $\hat{\boldsymbol{\eta}}$, to derive bootstrap responses

$$y_i^* = \hat{\boldsymbol{\eta}}^T \mathbf{u}_i + e_i^* / \sqrt{w_i}$$

I can now compute the usual weighted least squares estimate of $\boldsymbol{\eta}$ from this bootstrap sample, say $\hat{\boldsymbol{\eta}}^*$, and repeat B times.

As an example, consider data on brain weight *BrainWt* in grams and body weight *BodyWt* in kilograms for sixty-two species of mammal. These data are described in Section 5.1 of [1] and can be analyzed in *Arc* by loading the file `brains.lsp`.

After transforming the predictors *BrainWt* and *BodyWt* to their natural logarithms to get the terms, a linear regression model for $\log(\mathit{BrainWt})$ on $\log(\mathit{BodyWt})$ with constant variance function seems reasonable, and is easily fit using *Arc*. Standard analysis suggests that errors are approximately normal, and so the usual inference techniques for the intercept, η_0, and the slope, η_1, should be appropriate. This example is therefore ideal for investigating whether bootstrap methods can reinforce conclusions arrived at using normal theory.

When this model is fit using *Arc*, a *regression object* is created that has the same name as its menu, as described in [1]. A typical name for the object will be `L1`.

To begin a bootstrap analysis (assuming that `boot.lsp` has been loaded) of the distribution of the coefficient estimates using $B = 1000$ simulations by resampling (centered, modified) residuals, type the following in the text window's command line following the > prompt:

```
> (def r (send L1 :bootstrap :nboots 1000 :method :r-boot))
```

This creates a *bootstrap object* `r` that will use the "resampling residuals" paradigm, and has B=1000. To change B to another number, just replace the `1000`, and to use the "resampling cases" paradigm, just change the `:r-boot` to `:c-boot`. I can now send `r` various messages to produce plots or calculate quantities.

Like other object in *Arc*, typed commands can be used to get information about the bootstrap object. For example, typing

```
> (send r :histograms)
```

produces a histogram of the bootstrap intercept estimates, $\hat{\eta}_0^*$. The slidebar in the lower left corner of the plot controls which coefficient estimate to display—click once to the right of the slider to change the histogram to one for the bootstrap slope estimate, $\hat{\eta}_1^*$. The two histograms are shown in Figure 1 with superimposed Gaussian kernel density smooths indicating near-normality.

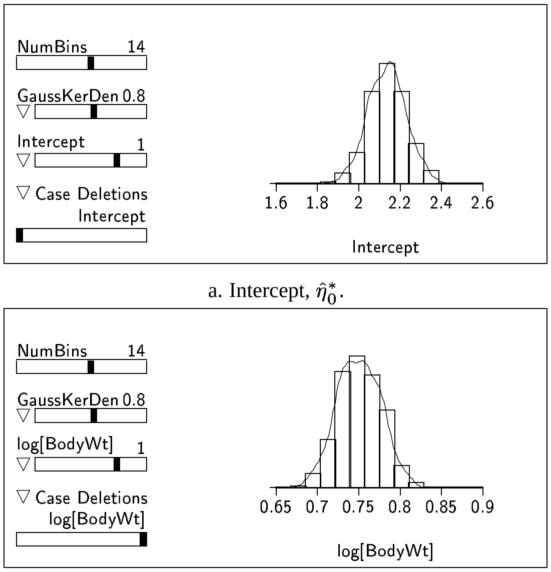
Similarly, typing

```
> (send r :probability-plots)
```

produces a probability plot (also known as a QQ-plot) of the *studentized* bootstrap intercept estimates, with a t distribution on $n - p$ degrees of freedom (df) as the reference distribution. (Recall that when sampling normal data, estimates using the usual variance estimate rather than the true variance are t distributed rather than normally distributed.) The studentized bootstrap slope estimates are defined to be

$$\frac{\hat{\eta}_0^* - \hat{\eta}_0}{\mathrm{se}(\hat{\eta}_0^*)}$$

where $\hat{\eta}_0^*$ is the bootstrap estimate of η_0, $\hat{\eta}_0$ is the usual estimate of η_0 from the original sample, and $\mathrm{se}(\hat{\eta}_0^*)$ is the bootstrap estimate of the standard error of $\hat{\eta}_0$, that is the



a. Intercept, $\hat{\eta}_0^*$.

b. Slope, $\hat{\eta}_1^*$.

Figure 1: Histograms of bootstrap coefficient estimates for the `brains.lsp` data set.

standard deviation of the $\hat{\eta}_0^*$'s in the bootstrap samples. Note that this definition comes from the "$\hat{\eta}^*$ is to $\hat{\eta}$ as $\hat{\eta}$ is to η" idea mentioned in Section 2.

The slidebar in the lower left corner of the plot again controls which coefficient estimate to display—click once to the right of the slider to change the probability plot to one for the studentized bootstrap slope estimates. The two probability plots are shown in Figure 2 with superimposed $y = x$ lines indicating near-normality (although there is a hint of non-normality for the studentized bootstrap slope estimates).

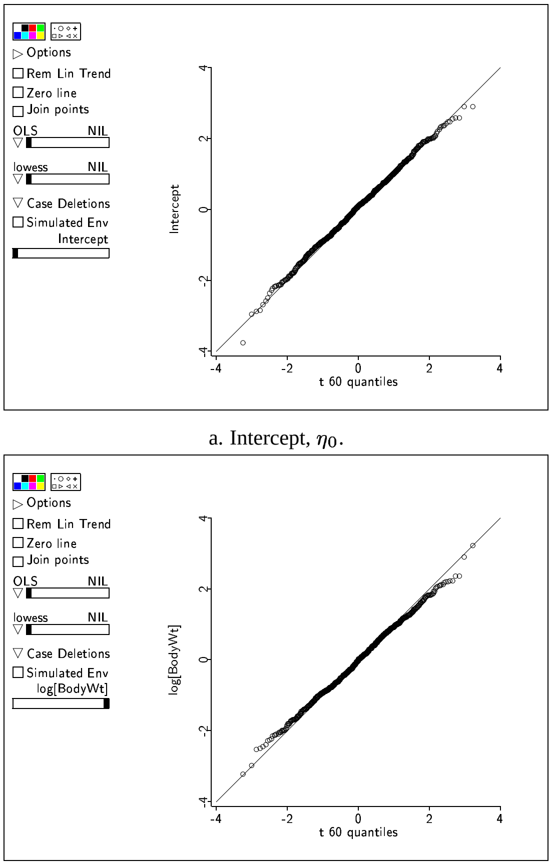
To display bootstrap estimates of the standard errors of the intercept and slope estimates, type

```
> (send r :bootstrap-standard-error)
Residual bootstrap
Coefficient    Std. Error
Intercept        0.094485
log[BodyWt]      0.026757
```

These standard error estimates compare with the usual (linear regression model theory) estimates of 0.0960 and 0.0285 respectively. And note that the usual estimates are also the *ideal* bootstrap estimates of standard error, that is the bootstrap estimates as $B \to \infty$ (adjusted by $[n/(n-k)]^{1/2}$).

To display bootstrap estimates of the bias for the intercept and slope estimates, type

```
> (send r :bootstrap-bias)
Residual bootstrap
Coefficient          Bias
Intercept        0.005359
log[BodyWt]     -0.000514
```



a. Intercept, η_0.

b. Slope, η_1.

Figure 2: Probability plots of studentized bootstrap coefficient estimates for the `brains.lsp` data set.

These bias estimates are reassuringly small since the usual coefficient estimates are unbiased. Note that bootstrap estimates of bias again come from the "$\hat{\eta}^*$ is to $\hat{\eta}$ as $\hat{\eta}$ is to η" idea mentioned in Section 2. In particular, the bootstrap estimate of bias for the intercept estimate is $\hat{\eta}_0^*(\cdot) - \hat{\eta}_0$, where $\hat{\eta}_0^*(\cdot)$ is the average of the $\hat{\eta}_0^*$'s.

To calculate normal theory confidence intervals for the coefficient estimates, together with bootstrap confidence intervals using both the percentile method and the BCa method, type

```
> (send r :display-confidence-intervals)
Residual bootstrap
Level  Normal Theory         Percentile Bootstrap   BCa Bootstrap
Coefficient: Intercept
0.95   (1.94267 2.3269)      (1.94677 2.32673)      (1.93574 2.31693)
Coefficient: log[BodyWt]
0.95   (0.694751 0.808622)   (0.698998 0.803802)    (0.700898 0.804064)
```

Normal theory confidence intervals are just the usual "estimate ± multiplier × standard error." Percentile bootstrap intervals use percentiles of the empirical distribution of the bootstrap estimates to estimate percentiles of the true distribution of the coefficients. BCa bootstrap intervals use an improved version of the percentile bootstrap method which tends to have better coverage properties. BCa intervals are discussed in Section 14.3 of [3].

The default level for the confidence interval calculations is 0.95. Type the following (before sending the `:display-confidence-intervals` message) to change the level to 0.9:

```
> (send r :levels .9)
```

or the following to allow intervals for both 0.90 and 0.95 confidence levels to be displayed together:

```
> (send r :levels '(.9 .95))
```

To calculate bootstrap confidence intervals using the percentile method for the studentized coefficient estimates, type

```
> (send r :stud-percentile-boot-intervals)
Residual bootstrap
Level   Percentile Bootstrap
Coefficient: Intercept
 0.9    (-1.55743 1.76993)
0.95    (-1.98995 2.03149)
Coefficient: log[BodyWt]
 0.9    (-1.66905 1.59792)
0.95    (-1.96914 1.94774)
```

The corresponding t quantiles on $n - k = 60$ df are 1.67065 and 2.0003 respectively. Thus, the percentile bootstrap intervals indicate that the resampling results agree closely with those obtained from standard methods.

3.2 Resampling cases

As described above, I want to try to estimate the distribution of $(y, \mathbf{x})$ by resampling cases. Thus, I sample from the integers running from 1 to n to get bootstrap indices

$$j_i^* \text{ randomly sampled with replacement from } (1, 2, \ldots, n)$$

I now need to derive bootstrap responses and predictor values from my bootstrap indices. Since, I am assuming that $(y, \mathbf{x})$ are sampled from a multivariate distribution $F(y, \mathbf{x})$, I derive bootstrap cases

$$(y_i^*, \mathbf{x}_i^*, w_i^*) = (y_{j_i^*}, \mathbf{x}_{j_i^*}, w_{j_i^*})$$

I can now compute the usual weighted least squares estimate of $\boldsymbol{\eta}$ from this bootstrap sample, and repeat B times. Applying this to the `brains.lsp` data set, I obtain:

```
> (def c (send L1 :bootstrap :nboots 1000 :method :c-boot))

> (send c :bootstrap-standard-error)
Case bootstrap
```

```
Coefficient    Std. Error
Intercept        0.093686
log[BodyWt]      0.023061

> (send c :bootstrap-bias)
Case bootstrap
Coefficient          Bias
Intercept       -0.001015
log[BodyWt]      0.000662

> (send c :display-confidence-intervals)
Case bootstrap
Level  Normal Theory         Percentile Bootstrap   BCa Bootstrap
Coefficient: Intercept
0.95   (1.94267 2.3269)      (1.95774 2.31821)      (1.96562 2.33279)
Coefficient: log[BodyWt]
0.95   (0.694751 0.808622)   (0.706941 0.796554)    (0.705292 0.793233)

> (send c :levels '(.9 .95))

> (send c :stud-percentile-boot-intervals)
Case bootstrap
Level   Percentile Bootstrap
Coefficient: Intercept
 0.9    (-1.59979 1.66336)
0.95    (-1.88978 1.95787)
Coefficient: log[BodyWt]
 0.9    (-1.54205 1.64522)
0.95    (-1.94031 1.94564)
```

The results are reassuringly similar to those obtained by resampling residuals.

Note that it is possible for case resampling to run into computational problems. In particular, given the numerical tolerances computers must work within, it is possible (albeit unlikely) for a bootstrap sample to produce a model with lower rank than the model for the original sample. The *Arc* bootstrap code does not currently check for this, and if it does happen, an unhelpful error message is returned and the bootstrap routine terminates. Future work will hopefully address this issue so that any lower rank bootstrap samples are ignored, allowing the routine to be completed.

4 Hypothesis tests for regression coefficients

Consider a hypothesis test with null hypothesis, NH, and test statistic, T. The p-value for this hypothesis test can be defined as the probability that a random variable with the same distribution as that of T when NH is true is as large or larger (in absolute value) than the observed value of T.

This suggests the following method for estimating a p-value using bootstrap ideas: figure out a null distribution for the data under NH and generate B bootstrap values of T under this distribution. The p-value estimate is then the proportion of times that the bootstrap T^*'s are as large or larger (in absolute value) than the (original sample) observed value of T. To apply this in practice however, I need to choose both a suitable T and a null distribution for the data under NH.

An alternative strategy uses a *pivot statistic* for T. Pivot statistics have distributions that do not depend on coefficient values, so there is no need to figure out a null distribution for the data under NH—I can just carry out my simulation under the alternative hypothesis, AH. Again, the p-value estimate is the proportion of times that the bootstrap T^*'s are as large or larger (in absolute value) than the (original sample) observed value of T.

4.1 Testing a single regression coefficient

The usual normal theory method for testing a single regression coefficient is a t-test of the studentized coefficient estimate.

As an alternative, one way to test a single regression coefficient using the pivot method outlined above is to use the pivot

$$T = \frac{\hat{\eta}_k - \eta_k}{\text{se}(\hat{\eta}_k)}$$

to test the k-th coefficient ($k = 0, \ldots, p-1$). Using the "$\hat{\eta}^*$ is to $\hat{\eta}$ as $\hat{\eta}$ is to η" idea mentioned in Section 2,

$$T^* = \frac{\hat{\eta}^*_k - \hat{\eta}_k}{\text{se}(\hat{\eta}^*_k)}$$

The (two-tailed) p-value estimate for $\eta_k = 0$ is then the proportion of times that the bootstrap T^*'s are as large or larger (in absolute value) than the (original sample) observed value of T,

$$T_0 = \frac{\hat{\eta}_k}{\text{se}(\hat{\eta}_k)}$$

This procedure can be performed in *Arc* using either residual resampling or case resampling. As an example, consider data from 1973 on accident rate, *Rate*, access points per mile, *Acpt*, truck volume percentage, *Trks*, average daily traffic count, *ADT*, length of segment, *Len*, signalized interchanges per mile, *Sigs*, and speed limit, *Slim*, for thirty-nine segments of Minnesota highways. These data are described in Section 11.3 of [1] and can be analyzed in *Arc* by loading the file `highway.lsp`.

A reasonably good linear regression model has the following mean function

$$\begin{aligned} \text{E}(\log_2(\textit{Rate})|\mathbf{x}) &= \eta_0 + \eta_1\log_2(\textit{Acpt}) + \eta_2\log_2(\textit{Trks}) + \eta_3\log_2(\textit{ADT}) \\ &\quad + \eta_4\log_2(\textit{Len}) + \eta_5\textit{Sigs1}^{-1} + \eta_6\textit{Slim} \qquad (5) \end{aligned}$$

where $\textit{Sigs1} = \textit{Sigs} + 1$, and constant variance function. (*Sigs* had many zero values, and so was re-centered to allow the possibility of subsequent transformation.) To separately test each mean function coefficient against zero, type the following:

```
> (def r (send L1 :bootstrap :nboots 1000 :method :r-boot))
> (send r :stud-boot-pvalues)
Residual bootstrap
Coefficient       p-value
Intercept          0.0000
log2[Acpt]         0.2210
log2[Trks]         0.2550
```

Table 1: Usual (two-tailed) t-test p-values for the highway accident data.

Coefficient	p-value
Intercept	0.0003
$\log_2$[Acpt]	0.2200
$\log_2$[Trks]	0.2530
$\log_2$[ADT]	0.3686
$\log_2$[Len]	0.0053
Sigs1^{-1}	0.1458
Slim	0.0484

```
log2[ADT]          0.3590
log2[Len]          0.0020
Sigs1^-1           0.1280
Slim               0.0400
> (def c (send L1 :bootstrap :nboots 1000 :method :c-boot))
> (send c :stud-boot-pvalues)
Case bootstrap
Coefficient       p-value
Intercept          0.0010
log2[Acpt]         0.1960
log2[Trks]         0.2130
log2[ADT]          0.3260
log2[Len]          0.0110
Sigs1^-1           0.1270
Slim               0.0450
```

where `L1` is the name of the linear regression model with mean function (5) and constant variance function. For comparison, the usual (two-tailed) t-test p-values are given in Table 1.

4.2 Testing a subset of mean function coefficients

Consider the following two models stated as hypotheses:

$$\begin{array}{ll} \text{NH:} & \mathrm{E}(y|\mathbf{x}) = \boldsymbol{\eta}_1^T \mathbf{u}_1 \text{ with } \mathrm{Var}(y|\mathbf{x}) = \sigma^2/w \\ \text{AH:} & \mathrm{E}(y|\mathbf{x}) = \boldsymbol{\eta}_1^T \mathbf{u}_1 + \boldsymbol{\eta}_2^T \mathbf{u}_2 \text{ with } \mathrm{Var}(y|\mathbf{x}) = \sigma^2/w \end{array}$$

where the vectors of predictors and mean function coefficients have been similarly partitioned into $\mathbf{u}^T = (\mathbf{u}_1, \mathbf{u}_2)^T$ and $\boldsymbol{\eta}^T = (\boldsymbol{\eta}_1, \boldsymbol{\eta}_2)^T$. Let $\mathbf{U}_1$ be the matrix consisting of the n sample row vectors $\mathbf{u}_1^T$ stacked on top of each other. Let $\mathbf{U}_2$ be defined similarly.

Testing these hypotheses is equivalent to testing $\boldsymbol{\eta}_2 = \mathbf{0}$ and the usual normal theory method for this is an F-test of the ratio

$$F = \frac{(RSS_{NH} - RSS_{AH})/(\mathrm{df}_{NH} - \mathrm{df}_{AH})}{\hat{\sigma}^2}$$

where *RSS* indicates residual sum of squares from the model fit and $\hat{\sigma}^2$ is the estimate of σ^2 from the fit of the AH model.

As an alternative, one way to test $\boldsymbol{\eta}_2 = \mathbf{0}$ using the pivot method outlined above is to choose the pivot

$$T = \frac{(\hat{\boldsymbol{\eta}}_2 - \boldsymbol{\eta}_2)^T(\mathbf{U}_{2.1}^T\mathbf{U}_{2.1})(\hat{\boldsymbol{\eta}}_2 - \boldsymbol{\eta}_2)}{\hat{\sigma}^2}$$

where $\mathbf{U}_{2.1} = (\mathbf{I} - \mathbf{U}_1(\mathbf{U}_1^T\mathbf{U}_1)^{-1}\mathbf{U}_1^T)\mathbf{U}_2$, the part of $\mathbf{U}_2$ with the linear effect of $\mathbf{U}_1$ subtracted out. Note that the numerator of T is just another way of writing $RSS_{NH} - RSS_{AH}$.

Using the "$\hat{\eta}^*$ is to $\hat{\eta}$ as $\hat{\eta}$ is to η" idea mentioned in Section 2,

$$T^* = \frac{(\hat{\boldsymbol{\eta}}_2^* - \hat{\boldsymbol{\eta}}_2)^T(\mathbf{U}_{2.1}^T\mathbf{U}_{2.1})(\hat{\boldsymbol{\eta}}_2^* - \hat{\boldsymbol{\eta}}_2)}{\hat{\sigma}^{*2}}$$

The *p*-value estimate is then the proportion of times that the bootstrap T^*'s are as large or larger than the (original sample) observed value of T,

$$T_0 = \frac{\hat{\boldsymbol{\eta}}_2^T(\mathbf{U}_{2.1}^T\mathbf{U}_{2.1})\hat{\boldsymbol{\eta}}_2}{\hat{\sigma}^2}$$

To actually compute T^* and T_0, note that

$$\begin{aligned}
\hat{\boldsymbol{\eta}}_2 &= (\mathbf{U}_{2.1}^T\mathbf{U}_{2.1})^{-1}\mathbf{U}_{2.1}^T\mathbf{y} \\
\hat{\boldsymbol{\eta}}_2^* &= (\mathbf{U}_{2.1}^T\mathbf{U}_{2.1})^{-1}\mathbf{U}_{2.1}^T\mathbf{y}^* \\
\hat{\boldsymbol{\eta}}_2^* - \hat{\boldsymbol{\eta}}_2 &= (\mathbf{U}_{2.1}^T\mathbf{U}_{2.1})^{-1}\mathbf{U}_{2.1}^T(\mathbf{y}^* - \mathbf{y})
\end{aligned}$$

where $\mathbf{y}$ is the vector of responses. Thus

$$\begin{aligned}
T^* &= \frac{(\mathbf{y}^* - \mathbf{y})^T\mathbf{U}_{2.1}(\mathbf{U}_{2.1}^T\mathbf{U}_{2.1})^{-1}\mathbf{U}_{2.1}^T(\mathbf{y}^* - \mathbf{y})}{\hat{\sigma}^{*2}} \\
T_0 &= \frac{\mathbf{y}^T\mathbf{U}_{2.1}(\mathbf{U}_{2.1}^T\mathbf{U}_{2.1})^{-1}\mathbf{U}_{2.1}^T\mathbf{y}}{\hat{\sigma}^2}
\end{aligned}$$

T^* and T_0 can therefore be computed with full model (AH) simulation only, and without the need to actually calculate the bootstrap coefficient estimates.

This procedure can be performed in *Arc* only using residual resampling; case resampling is inappropriate here. As an example, consider the highway accident data again.

First, I can test the mean function coefficients singly (as I did in Section 4.1). To test $\eta_1 = 0$ in mean function (5), type the following:

```
> (send L1 :sigtest-boot-pvalue L2 :nboots 1000 :method :r-boot)
Residual bootstrap
p-value: 0.2290
```

where `L1` is the name of the linear regression model with mean function (5) and `L2` is the name of the linear regression model with mean function

$$\begin{aligned}
\mathrm{E}(\log_2(\mathit{Rate})|\mathbf{x}) &= \eta_0 + \eta_2\log_2(\mathit{Trks}) + \eta_3\log_2(\mathit{ADT}) \\
&\quad +\eta_4\log_2(\mathit{Len}) + \eta_5\mathit{Sigs1}^{-1} + \eta_6\mathit{Slim}
\end{aligned}$$

Table 2: Bootstrap subset test p-values for the highway accident data.

Coefficient	p-value
$\log_2$[Acpt]	0.2290
$\log_2$[Trks]	0.2220
$\log_2$[ADT]	0.3720
$\log_2$[Len]	0.0050
Sigs1^{-1}	0.1380
Slim	0.0450

Note that the syntax used here is different to that used in previous examples; the method `:sigtest-boot-pvalue` is sent directly to the full model (AH) with the submodel (NH) as an argument. There is no preliminary step of setting up a bootstrap object `r` for this method.

I can test the other mean function coefficients singly in a similar way. The p-value estimates I obtained are given in Table 2.

I can test a subset of more than one mean function coefficients in a similar way. For example, to test $\eta_5 = \eta_6 = 0$ in mean function (5), type the following:

```
> (send L1 :sigtest-boot-pvalue L3 :nboots 1000 :method :r-boot)
Residual bootstrap
p-value: 0.0260
```

where `L1` is the name of the linear regression model with mean function (5) and `L3` is the name of the linear regression model with mean function

$$\begin{aligned} \mathrm{E}(\log_2(\mathit{Rate})|\mathbf{x}) &= \eta_0 + \eta_1 \log_2(\mathit{Acpt}) + \eta_2 \log_2(\mathit{Trks}) \\ &\quad + \eta_3 \log_2(\mathit{ADT}) + \eta_4 \log_2(\mathit{Len}) \end{aligned}$$

For comparison, the usual F-test p-value for this is 0.027.

5 Further Examples

I provide examples of the procedures discussed above for two further data sets. Section 5.1 considers a model that includes weights to demonstrate how the procedures cope with non-constant weights. Section 5.2 considers a data set with missing values on some variables to demonstrate how the procedures cope with missing data.

5.1 Particle Physics Example

Consider data on an experiment in particle physics, where x is the inverse of the total input energy, y measures the total output of particles, and S is the known population standard deviations of $y|x$ for each value of x. These data are described in Section 9.2 of [1] and can be analyzed in *Arc* by loading the file `physics.lsp`.

A reasonably good linear regression model has a quadratic polynomial mean function, fit with weights equal to S^{-2}; call this model `L1`. I then carry out the following bootstrap procedures.

```
> (def r (send L1 :bootstrap :nboots 1000 :method :r-boot))
> (send r :bootstrap-standard-error)
Residual bootstrap
Coefficient    Std. Error
Intercept        6.428289
x               85.642674
x^2            253.673145

> (def c (send L1 :bootstrap :nboots 1000 :method :c-boot))
> (send c :bootstrap-standard-error)
Case bootstrap
Coefficient    Std. Error
Intercept       15.753806
x              164.532466
x^2            438.831268
```

These standard error estimates compare with the usual (linear regression model theory) estimates of 6.46, 85.37, and 250.59 respectively. The differences between residual and case resampling probably reflect the relatively small sample size of ten for this example. An assumption of independence between the errors and the predictors becomes easier to justify (if it is true) the larger the sample size. In this case, the small sample size makes it quite hard to justify this assumption, and this "extra uncertainty" is reflected in the larger bootstrap standard errors for case resampling (which does not make this assumption).

```
> (send r :bootstrap-bias)
Residual bootstrap
Coefficient          Bias
Intercept       -0.067248
x                1.841999
x^2             -5.894250

> (send c :bootstrap-bias)
Case bootstrap
Coefficient          Bias
Intercept       -3.084323
x               20.723333
x^2            -18.050084
```

These bias estimates should be considered relative to the usual (linear regression model theory) coefficient estimates of 183.83, 0.97, and 1597.50 respectively (and taking into account the standard error estimates also). Again, the case resampling estimates are larger (in absolute value) than the residual resampling estimates.

```
> (send r :display-confidence-intervals)
Residual bootstrap
Level  Normal Theory         Percentile Bootstrap   BCa Bootstrap
Coefficient: Intercept
0.95   (168.557 199.104)     (171.873 196.153)      (174.422 200.135)
Coefficient: x
0.95   (-200.894 202.836)    (-163.559 161.414)     (-220.236 129.548)
Coefficient: x^2
0.95   (1004.96 2190.05)     (1130.02 2095.22)      (1198.33 2218.94)

> (send c :display-confidence-intervals)
```

Table 3: Usual (two-tailed) t-test p-values for the physics data.

Coefficient	p-value
Intercept	0.0000
x	0.9915
x^2	0.0004

```
Case bootstrap
Level  Normal Theory         Percentile Bootstrap   BCa Bootstrap
Coefficient: Intercept
0.95   (168.557 199.104)     (142.923 199.533)      (164.268 203.067)
Coefficient: x
0.95   (-200.894 202.836)    (-282.78 374.272)      (-368.806 185.637)
Coefficient: x^2
0.95   (1004.96 2190.05)     (761.872 2639.3)       (1167.75 2917.93)
```

The confidence interval estimates also reflect the standard error and bias issues discussed above.

```
> (send r :stud-boot-pvalues)
Residual bootstrap
Coefficient       p-value
Intercept          0.0000
x                  0.9930
x^2                0.0000

> (send c :stud-boot-pvalues)
Case bootstrap
Coefficient       p-value
Intercept          0.0000
x                  0.9780
x^2                0.0010
```

For comparison, the usual (two-tailed) t-test p-values are given in Table 3.

Finally, I test the coefficient for x equal to zero an alternative way, that is with the bootstrap subset test:

```
> (send L1 :sigtest-boot-pvalue L2 :nboots 1000 :method :r-boot)
Residual bootstrap
p-value: 0.9320
```

where `L2` is the name of the linear regression model with mean function containing only an intercept term and x^2.

5.2 Sleep Example

Consider data on sixty-two mammal species, where *TS* is total sleep per day, *BW* is body weight, *BrW* is brain weight, *Life* is maximum life span, *GP* is gestation period, and D_1 is a danger index. Not all variables are measured on each species. These data are described in Section 10.4 of [1] and can be analyzed in *Arc* by loading the file `sleep.lsp`.

A reasonably good linear regression model has the following mean function

$$\begin{aligned}\mathrm{E}(TS|\mathbf{x}) &= \eta_0 + \eta_1 \log_2(BW) + \eta_2 \log_2(BrW) + \eta_3 \log_2(Life) \\ &\quad + \eta_4 \log_2(GP) + \eta_5 D_1 \end{aligned} \tag{6}$$

and constant variance function. Let `L1` be the name of the linear regression model with mean function (6) and constant variance function. I then carry out the following bootstrap procedures.

```
> (def r (send L1 :bootstrap :nboots 1000 :method :r-boot))
> (send r :bootstrap-standard-error)
Residual bootstrap
Coefficient    Std. Error
Intercept        2.607657
log2[BW]         0.348886
log2[BrW]        0.526081
log2[Life]       0.527639
log2[GP]         0.490668
D1               0.950746

> (def c (send L1 :bootstrap :nboots 1000 :method :c-boot))
> (send c :bootstrap-standard-error)
Case bootstrap
Coefficient    Std. Error
Intercept        3.183498
log2[BW]         0.372109
log2[BrW]        0.625351
log2[Life]       0.779163
log2[GP]         0.585363
D1               0.967958
```

These standard error estimates compare with the usual (linear regression model theory) estimates of 2.66, 0.34, 0.53, 0.53, 0.49, and 0.91 respectively. The differences between residual and case resampling again possibly reflect a failure of the error distribution assumption. There are larger bootstrap standard errors for case resampling (which does not make this assumption) than for residual resampling (which does make this assumption).

```
> (send r :bootstrap-bias)
Residual bootstrap
Coefficient          Bias
Intercept       -0.032786
log2[BW]        -0.020547
log2[BrW]        0.032676
log2[Life]      -0.019615
log2[GP]        -0.000090
D1               0.060512

> (send c :bootstrap-bias)
Case bootstrap
Coefficient          Bias
Intercept        0.100382
log2[BW]        -0.047133
```

```
log2[BrW]        0.127983
log2[Life]      -0.121676
log2[GP]        -0.034941
D1               0.051909
```

These bias estimates should be considered relative to the usual (linear regression model theory) coefficient estimates of 19.31, 0.02, -0.55, 0.43, -0.97, and -3.82 respectively (and taking into account the standard error estimates also). Again, the case resampling estimates are generally larger (in absolute value) than the residual resampling estimates.

```
> (send r :display-confidence-intervals)
Residual bootstrap
Level  Normal Theory         Percentile Bootstrap   BCa Bootstrap
Coefficient: Intercept
0.95   (13.9479 24.6703)     (13.9825 24.4952)      (14.4963 24.8782)
Coefficient: log2[BW]
0.95   (-0.669011 0.714042)  (-0.687735 0.691718)   (-0.643718 0.705958)
Coefficient: log2[BrW]
0.95   (-1.61161 0.51011)    (-1.5596 0.531056)     (-1.66774 0.442263)
Coefficient: log2[Life]
0.95   (-0.651064 1.50249)   (-0.584592 1.43294)    (-0.531056 1.51739)
Coefficient: log2[GP]
0.95   (-1.95049 0.00792257) (-1.92356 -0.0262216)  (-1.9221 -0.0262216)
Coefficient: D1
0.95   (-5.64703 -1.99287)   (-5.65607 -1.91845)    (-5.77184 -2.04087)

> (send c :display-confidence-intervals)
Case bootstrap
Level  Normal Theory         Percentile Bootstrap   BCa Bootstrap
Coefficient: Intercept
0.95   (13.9479 24.6703)     (12.4558 25.025)       (12.1751 24.8137)
Coefficient: log2[BW]
0.95   (-0.669011 0.714042)  (-0.748034 0.727619)   (-0.615866 0.802303)
Coefficient: log2[BrW]
0.95   (-1.61161 0.51011)    (-1.52022 1.13081)     (-1.73165 0.597859)
Coefficient: log2[Life]
0.95   (-0.651064 1.50249)   (-1.58862 1.62901)     (-1.41923 1.69452)
Coefficient: log2[GP]
0.95   (-1.95049 0.00792257) (-2.10444 0.15314)     (-2.00926 0.254512)
Coefficient: D1
0.95   (-5.64703 -1.99287)   (-5.68968 -1.82073)    (-5.81219 -2.1066)
```

The confidence interval estimates also reflect the standard error and bias issues discussed above.

```
> (send r :stud-boot-pvalues)
Residual bootstrap
Coefficient       p-value
Intercept          0.0000
log2[BW]           0.9450
log2[BrW]          0.2980
log2[Life]         0.4480
log2[GP]           0.0420
D1                 0.0000
```

Table 4: Usual (two-tailed) t-test p-values for the sleep data.

Coefficient	p-value
Intercept	0.0000
$\log_2$[BW]	0.9477
$\log_2$[BrW]	0.3011
$\log_2$[Life]	0.4302
$\log_2$[GP]	0.0518
D1	0.0001

```
> (send c :stud-boot-pvalues)
Case bootstrap
Coefficient       p-value
Intercept          0.0000
log2[BW]           0.9390
log2[BrW]          0.2540
log2[Life]         0.3730
log2[GP]           0.0410
D1                 0.0000
```

For comparison, the usual (two-tailed) t-test p-values are given in Table 4.

Finally, I test the coefficients for $\log_2(BW)$ and $\log_2(BrW)$ both equal to zero with the bootstrap subset test:

```
> (send L1 :sigtest-boot-pvalue L2 :nboots 1000 :method :r-boot)
Residual bootstrap
p-value: 0.1610
```

where `L2` is the name of the linear regression model with mean function

$$\mathrm{E}(TS|\mathbf{x}) \quad = \quad \eta_0 + \eta_3 \log_2(Life) + \eta_4 \log_2(GP) + \eta_5 D_1$$

For comparison, the usual F-test p-value for this is 0.1055.

6 Generalizations

Some other areas in linear regression for which bootstrap methods have potential include prediction and variable selection. Also, the methods discussed in this report can be extended to work with generalized linear models and nonlinear models. The computer routines in `boot.lsp` can provide a starting point for these other methods.

References

[1] Cook, R.D., and Weisberg, S. (1999), *Applied Regression Including Computing and Graphics*, New York: Wiley.

[2] Davison, A.C., and Hinkley, D.V. (1997), *Bootstrap Methods and their Application*, New York: Cambridge University Press.

[3] Efron, B., and Tibshirani, R.J. (1993), *An Introduction to the Bootstrap*, New York: Chapman and Hall.

[4] Tierney, L. (1990), *Lisp-Stat: An Object-Oriented Environment for Statistical Computing and Dynamic Graphics*, New York: Wiley.